MAVEN PROJECT TEMPLATES

Maven provides users,a very large list of different types of project templates (614 in numbers) using concept of **Archetype**. Maven helps users to quickly start a new java project using following command

```
mvn archetype:generate
```

What is Archetype?

Archetype is a Maven plugin whose task is to create a project structure as per its template. We are going to use *quickstart* archetype plugin to create a simple java application here.

Using Project Template

Let's open command console, go the **C:\ > MVN** directory and execute the following **mvn** command

```
C:\MVN>mvn archetype:generate
```

Maven will start processing and will ask to choose required archetype

```
INFO] Scanning for projects...
[INFO] Searching repository for plugin with prefix: 'archetype'.
[INFO] -------------------------------------------------------------------
[INFO] Building Maven Default Project
[INFO]    task-segment: [archetype:generate] (aggregator-style)
[INFO] -------------------------------------------------------------------
[INFO] Preparing archetype:generate
...
600: remote -> org.trailsframework:trails-archetype (-)
601: remote -> org.trailsframework:trails-secure-archetype (-)
602: remote -> org.tynamo:tynamo-archetype (-)
603: remote -> org.wicketstuff.scala:wicket-scala-archetype (-)
604: remote -> org.wicketstuff.scala:wicketstuff-scala-archetype
Basic setup for a project that combines Scala and Wicket,
depending on the Wicket-Scala project.
Includes an example Specs test.)
605: remote -> org.wikbook:wikbook.archetype (-)
606: remote -> org.xaloon.archetype:xaloon-archetype-wicket-jpa-glassfish (-)
607: remote -> org.xaloon.archetype:xaloon-archetype-wicket-jpa-spring (-)
608: remote -> org.xwiki.commons:xwiki-commons-component-archetype
(Make it easy to create a maven project for creating XWiki Components.)
609: remote -> org.xwiki.rendering:xwiki-rendering-archetype-macro
(Make it easy to create a maven project for creating XWiki Rendering Macros.)
610: remote -> org.zkoss:zk-archetype-component (The ZK Component archetype)
611: remote -> org.zkoss:zk-archetype-webapp (The ZK wepapp archetype)
612: remote -> ru.circumflex:circumflex-archetype (-)
613: remote -> se.vgregion.javg.maven.archetypes:javg-minimal-archetype (-)
614: remote -> sk.seges.sesam:sesam-annotation-archetype (-)
Choose a number or apply filter
(format: [groupId:]artifactId, case sensitive contains): 203:
```

Press Enter to choose to default option(203: maven-archetype-quickstart)

Maven will ask for particular version of archetype

```
Choose org.apache.maven.archetypes:maven-archetype-quickstart version:
1: 1.0-alpha-1
2: 1.0-alpha-2
3: 1.0-alpha-3
4: 1.0-alpha-4
```

```
5: 1.0
6: 1.1
Choose a number: 6:
```

Press Enter to choose to default option(6: maven-archetype-quickstart:1.1)

Maven will ask for project details. Enter project details as asked.Press Enter if default value is provided. You can override them by entering your own value.

```
Define value for property 'groupId': : com.companyname.insurance
Define value for property 'artifactId': : health
Define value for property 'version': 1.0-SNAPSHOT:
Define value for property 'package': com.companyname.insurance:
```

Maven will ask for project details confirmation. Press enter or press Y

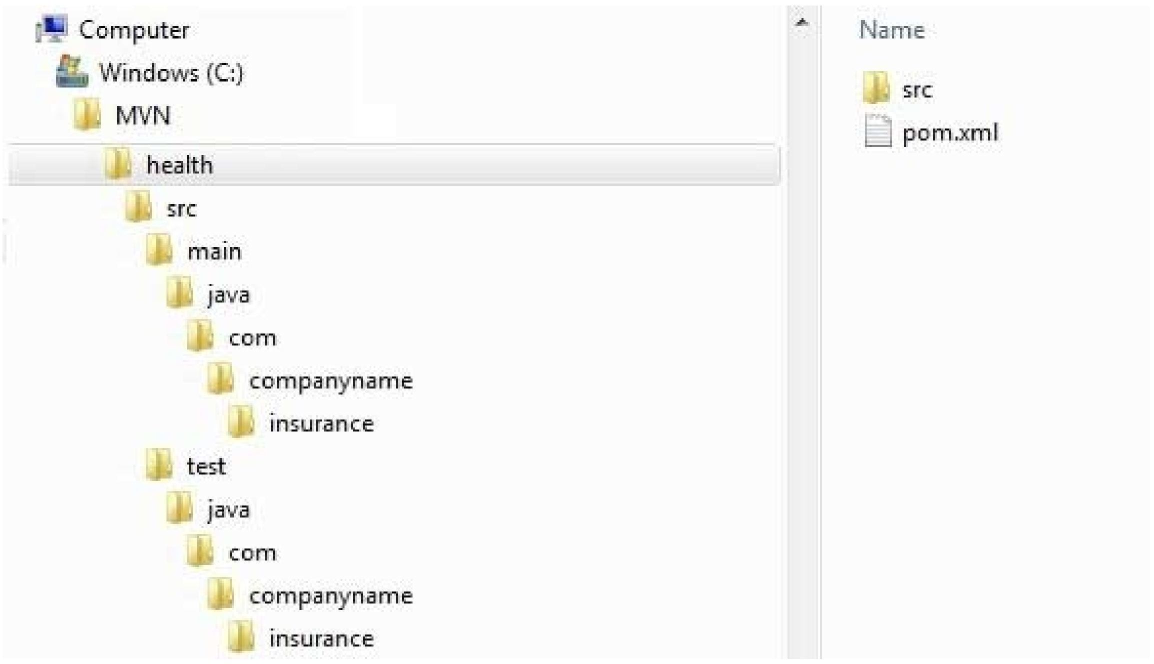
```
Confirm properties configuration:
groupId: com.companyname.insurance
artifactId: health
version: 1.0-SNAPSHOT
package: com.companyname.insurance
Y:
```

Now Maven will start creating project structure and will display the following:

```
[INFO] -----------------------------------------------------------------------
[INFO] Using following parameters for creating project
from Old (1.x) Archetype: maven-archetype-quickstart:1.1
[INFO] -----------------------------------------------------------------------
[INFO] Parameter: groupId, Value: com.companyname.insurance
[INFO] Parameter: packageName, Value: com.companyname.insurance
[INFO] Parameter: package, Value: com.companyname.insurance
[INFO] Parameter: artifactId, Value: health
[INFO] Parameter: basedir, Value: C:\MVN
[INFO] Parameter: version, Value: 1.0-SNAPSHOT
[INFO] project created from Old (1.x) Archetype in dir: C:\MVN\health
[INFO] -----------------------------------------------------------------------
[INFO] BUILD SUCCESSFUL
[INFO] -----------------------------------------------------------------------
[INFO] Total time: 4 minutes 12 seconds
[INFO] Finished at: Fri Jul 13 11:10:12 IST 2012
[INFO] Final Memory: 20M/90M
[INFO] -----------------------------------------------------------------------
```

Created Project

Now go to **C:\ > MVN** directory. You'll see a java application project created named **health** which was given as *artifactId* at the time of project creation. Maven will create a standard directory layout for the project as shown below:



Created POM.xml

Maven generates a POM.xml file for the project as listed below:

```
<project xmlns="http://maven.apache.org/POM/4.0.0"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
   http://maven.apache.org/xsd/maven-4.0.0.xsd">
   <modelVersion>4.0.0</modelVersion>
   <groupId>com.companyname.insurance</groupId>
   <artifactId>health</artifactId>
   <version>1.0-SNAPSHOT</version>
   <packaging>jar</packaging>
   <name>health</name>
   <url>http://maven.apache.org</url>
   <properties>
      <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
   </properties>
   <dependencies>
      <dependency>
      <groupId>junit</groupId>
         <artifactId>junit</artifactId>
         <version>3.8.1</version>
         <scope>test</scope>
      </dependency>
   </dependencies>
</project>
```

Created App.java

Maven generates sample java source file, App.java for the project as listed below:

Location: **C:\ > MVN > health > src > main > java > com > companyname > insurance > App.java**

```
package com.companyname.insurance;

/**
 * Hello world!
 *
 */
public class App
{
    public static void main( String[] args )
    {
        System.out.println( "Hello World!" );
    }
}
```

Created AppTest.java

Maven generates sample java source test file, AppTest.java for the project as listed below:

Location: **C:\ > MVN > health > src > test > java > com > companyname > insurance > AppTest.java**

```
package com.companyname.insurance;

import junit.framework.Test;
import junit.framework.TestCase;
import junit.framework.TestSuite;

/**
 * Unit test for simple App.
 */
public class AppTest
    extends TestCase
{
    /**
     * Create the test case
     *
     * @param testName name of the test case
     */
    public AppTest( String testName )
    {
        super( testName );
    }

    /**
     * @return the suite of tests being tested
     */
    public static Test suite()
    {
        return new TestSuite( AppTest.class );
    }

    /**
     * Rigourous Test :-)
     */
    public void testApp()
    {
        assertTrue( true );
    }
}
```

That's it. Now you can see the power of Maven. You can create any kind of project using single command in maven and can kick-start your development.